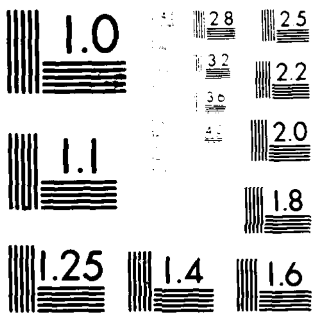
AD-A112 974 STANFORD UNIV CA DEPT OF COMPUTER SCIENCE F/G 12/1
THE ADDITIVE AND LOGICAL COMPLEXITIES OF LINEAR AND BILINEAR AR--ETC(U)
JUN 81 V Y PAN N00014-81-K-0269
UNCLASSIFIED STAN-CS-81-859 NL

1 OF 1
AD A
112974

END
DATE
FILMED
04-82
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

June 1981

Report. No. STAN-CS-81-859

13

AD A112974

The Additive and Logical Complexities of Linear and Bilinear Arithmetic Algorithms

by

V. Ya. Pan

Research sponsored by

Office of Naval Research
National Science Foundation

Department of Computer Science

Stanford University
Stanford, CA 94305

LELAND STANFORD JUNIOR UNIVERSITY
ORGANIZED 1891

DTIC
ELECTE
APR 2 1982
S D
E

8

The Additive and Logical Complexities of Linear and Bilinear Arithmetic Algorithms

V. Ya. Pan[*]
Institute for Advanced Study
Princeton, New Jersey 08540
and
Computer Science Department
Stanford University
Stanford, California 94305

Abstract. It is shown that $C_A(\pm) + do(D(A)) + ir(D(A)) = \Omega(K+Q)\log(K+Q)$ and $C_A(\pm) + do(\tilde{D}(A)) = \Omega(K+Q)\log(K+Q)$ in the cases of DFT, vector convolution, and matrix mutliplication. Here $C_A(\pm)$ is the additive complexity of a (bi)linear algorithm A for the given problem, $D(A)$ and $\tilde{D}(A)$ are the two acyclic digraphs that represent A, each of them is obtained from another one by reversing directions of all edges, $ir(D)$ and $do(D)$ are two numbers that are introduced to measure the structural deficiencies of an acyclic digraph D. K and Q are the numbers of outputs and input-variables. $do(\tilde{D}(A))$, $do(D(A))$, and $ir(D(A))$ characterize the logical complexity of A. Also lower bounds on $C_A(\pm) + do(D(A))$ and on $C_A(\pm)$ are expressed in terms of algebraic quantities such as the ranks of matrices and of multidimensional tensors associated with the problems.

Accession For
NTIS GRA&I ☒
DTIC TAB ☐
Unannounced ☐
Justification per ltr on file
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A

DTIC COPY INSPECTED 3

[*] This research has been supported by National Science Foundation grant MSC-8003347, Institute for Advanced Study; by National Science Foundation grant MCS-77-23738 and Office of Naval Research contract N00014-81-K-0269, Stanford University. Reproduction in whole or in part is permitted for any purpose of the United States government.

1. Introduction.

The design and analysis of arithmetic algorithms for Discrete Fourier Transform, DFT, convolution of vectors, CV, cyclic convolution of vectors, CCV, and matrix multiplication, MM, are among the central problems of algebraic complexity theory, cf. [1–4]. In this paper we analyze the arithmetic and logical complexity of such algorithms. We focus on the classes of linear and bilinear arithmetic algorithms (see the definitions in Sections 2, 3 where we basically follow [1-4]). There are three reasons for that choice. First, the linear and bilinear algorithms are natural ones. It is easier to analyze and to implement them. Secondly, the fastest known arithmetic algorithms for DFT, CV, CCV, and MM are linear and bilinear ones. Thirdly, it is known that all arithmetic algorithms for a linear or bilinear arithmetic computational problems such as DFT, CV, CCV, and MM can be turned into linear or respectively bilinear algorithms for the same problem and the total number of arithmetic operations in the algorithm remains the same up to a constant factor, cf. [3, pp. 34–38, 117].

For the analysis of the arithmetic complexity of linear and bilinear algorithms, it is technically convenient to estimate separately $C(\pm)$, $C(^*)$, and C_{sc}, the total numbers of additions/subtractions, nonscalar multiplications, and scalar multiplications respectively involved in the algorithm. This can be considered the first step of the study followed by the estimates for the bit-complexity of the algorithms, cf. [5]. So far the further progress in the area has been blocked by the two following problems. Estimate

i) $C_{\min}(^*)$ for MM (cf. [6,7]),

and

ii) $C_{\min}(\pm)$ for DFT, CV, and CCV (cf. [4, 8-11]).

In this paper we seek for lower bounds on $C_{\min}(\pm)$, the additive complexity of a given (bi)linear problem, that is on the minimum $C_A(\pm)$ over all (bi)linear algorithms A for that problem. (This also had led us to the study of logical complexity of algorithms.) We focus on the cases of DFT, CV, CCV, and MM.

Despite the long and intensive study of the problem (cf. [1–4]), all known lower bounds on $C_{\min}(\pm)$ still rely on the methods of substitution due to [12–14]; cf. also [15], consequently the lower bounds range between K and $K+Q$. Here Q and K are the total numbers of input variables and respectively of (linearly independent) outputs of the algorithm. However, those methods themselves by their nature cannot give even the bound $K+Q+1$. For comparison re[illegible]t the best known upper bounds on $C(\pm)$ have the asymptotic order $(K+Q)\log(K$ [illegible] in the cases of DFT, CV, and CCV, cf. [8,9], and $(K+Q)^\alpha$, $\alpha \approx 5/4$, in the case of MM, cf. [6,7] (α in [7] is less than [6] by ≈ 0.01 but the approach of [7] substantially uses the earlier designs whose origin is surveyed in detail in [6] while in the otherwise successful paper [7] the confusing name, "Schönhage's construction", does not help the reader).

So far the lower bound $K+Q+1$ on the $C(\pm)$ has been obtained for neither of the problems (DFT, CV, CCV, MM) unless the model of computation is seriously restricted as in [16] where the bound $\Omega(K+Q)\log(K+Q)$ was established by an ingenious method but only under quite a strong requirement that the constants of the algorithms be bounded by 1.

In this paper we do not impose any restrictions on our (bi)linear model of computation. In Section 5.1–5.3 we associate (bi)linear algorithms A with acyclic digraphs $D=D(A)$. Their vertices have outdegrees 2 and 0 and represent the $\pm$ operations of A and the input-variables of A respectively. Then $C_A(\pm)$ equals the number of vertices of D that have outdegree 2. In order to estimate that number, we try to partition all vertices of D into $p(D)$ levels ($p(D)$, the profundity of D, is defined in Section 5.2) whose cardinalities are bounded from below by r, the number of linearly independent outputs of the problem. Then

$$p(D)r \leq C_A(\pm).$$

However, the desirable partition might not exist. In that case we show that D must have some defects in its structure. To be more precise, we formally define $ir(D)+do(D)$, a measure for structural deficiency of an arbitrary acyclic digraph, $D=(V,E)$. We fix $L_0(D)$, a basic set of vertices of D. ($L_0(D)$ could be the set of all vertices of D of indegree 0 but for $D=D(A)$ we always choose $L_0(D)$ as the set of outputs of A.) Then we say that regularly all vertices of outdegree 0 (inputs of A) are equally (and hence maximally) removed from $L_0(D)$ (in terms of the longest paths from $L_0(D)$ to the vertices of outdegree 0). In the general case D can be irregular. Then we estimate $ir(D)$, the total deviation of vertices of outdegree 0 from those farthest from $L_0(D)$ positions. The $ir(D)$ measures the irregularity of D and will be called the irregularity number of D.

The disorder generated by an edge (u,v) from u to v, $u\neq v$, is defined as the difference between the positive lengths of the longest and the shortest directed paths from u to v. The disorder around v is the maximum (for all u) disorder generated by the edges (u,v) with the head v. The disorder number of D is the total disorder around all its vertices.

Both numbers, $ir(D)$ and $do(D)$ measure the deficiencies of the logical structure of the algorithm A represented by the digraph $D(A)$. (Their study might also be of interest itself.)

In Section 5.3 we show that both deficiencies can be corrected by a simple transformation of D. This transformation relies on joining some appropriate paths to D. In the result of the transformation all new vertices have outdegree 1 and other vertices do not change their outdegrees. If $D=D(A)$, such transformation basically consists in arranging $ir(D)+do(D)$ additional units to store some inputs and intermediate results of the algorithm A. We prove that totally $ir(D)+do(D)$ new vertices of outdegree 1 suffice to define a transformation of that kind that turns D into a new digraph $\overline{D}$ such that $ir(\overline{D}) = do(\overline{D}) = 0$ and $p(\overline{D}) = p(D)$. Then the desired partition of $\overline{D}$ into levels exists and defines the lower bound, $p(D)r$, on the number of vertices of the resulting digraph $\overline{D}$ that have nonzero outdegrees. Yet the latter number equals $C_A(\pm) + do(A) + ir(D(A))$ and can be considered the sum of additive and logical complexities of A. (The terms $do(A)$ and $ir(A)$ can be alternatively interpreted as the measures for additional storage.) In the cases of DFT, CV, CCV, and MM, $p(D)r = \Omega(K + Q)\log(K + Q)$ (see Theorem 5.3.1 and Equations (5.3.6)-(5.3.8) in Section 5.3). This defines nonlinear lower bounds on the $C_A(\pm) + do(D(A)) + ir(D(A))$ which are sharp up to a constant factor in the cases of DFT, CV, and CCV.

This narrows the class of possible choices for the designs of fast algorithms for the latter problems and also measures the sacrifices in the logical structure of such algorithms which are necessary if the algorithms exist.

In the remaining sections (5.4 and 6) we study the lower bounds on the $C_A(\pm) + do(D(A))$ and $C_A(\pm)$. In Section 5.4 we prove that $C_A(\pm)+do(D(A)) \geq \sum_{h=0}^{p(D)-1} r_{s(h)}(\mu)$ where $s(h)$ equals 2^h or $2^h - 1$, μ is the matrix of the coefficients of the given (bi)linear problem, and $r_p(\mu)$ is the p-rank of μ. (The p-rank of a matrix μ is defined in Section 4 as a generalization of the conventional rank, $r = r_0(\mu)$.) We conjecture (Proposition 4.1) that for some constant α, $0 < \alpha < 1$, we have that $r_p(\mu) = \Omega(K + Q)$ for $p = (K + Q)^\alpha$ in the cases where μ is the matrix associated with DFT, CV, CCV, and MM. We do not really attack that hard and interesting problem. Instead we establish other bounds on $r_p(\mu)$ and hence linear lower bounds on $C_A(\pm) + do(A)$ that exceed $K + Q$ (see Theorem 5.4.2 in Section 5). Also we prove (see Corollary 5.4.2 in Section 5.4) that at the very least any design of an algorithm A for DFT, CV, CCV, or MM such that $C_A(\pm) + do(A) = o(K + 0)\log(K + Q)$ must be a counterexample to our conjecture (that is to Proposition 4.1).

Finally we observe that we can reverse the directions of edges of $D(A)$ and choose the set of input-variables of A as the basic set $L_0(\tilde{D})$ of the new digraph, $\tilde{D}(A)$. Then our construction give the lower bound $C_A(\pm) + do(\tilde{D}(A)) = \Omega(K + Q)\log(K + Q)$ in the cases of DFT, CV, CCV, and MM.

In Section 6 we estimate $C_A(\pm)$ from below. (Assume that the increase of the disorder number of algorithms is not prohibitive.) We combine the approach of [16] with our extension of the concept of tensor ranks (cf. [1, p. 488] or [17], see also [18, 19]). We define $r(P(X))$, the rank of an arbitrary polynomial $P(X)$, and in particular, $r(m)$, the rank of any minor m of the matrix $\mu(X)$, that defines a bilinear computational problem. Then we show that

$$C_A(\pm) \geq \log_2 r(m)$$

for any bilinear algorithm A for such a problem and for any minor of $\mu(X)$. This leads to open problems of estimating $\max_m \log_2 r(m)$ or other related algebraic quantities. The material of Sections 2, 3, 5.1, 6 is self-contained.

2. Linear and bilinear arithmetic computational problems.

Hereafter we assume that all linear and bilinear forms that we consider are homogeneous, unless otherwise stated and that all logarithms are to the base 2. We use the following notation.

Notation. I, J, K, i, j, k are nonnegative integers, $\underline{\varphi}$ is a vector, μ is a matrix, τ is a (3-dimensional) tensor. $\det \mu$ is the determinant of μ. $(\mu)_{ij}$ is the entry of μ lying in row i and column j, $(\underline{\varphi})_i$ is the entry i of vector $\underline{\varphi}$. Similarly, the entries $(\tau)_{ijk}$ of tensor τ are defined. $\ell(\underline{\varphi})$, $b(\underline{\varphi})$ is a linear, respectively bilinear form of indeterminates $(\underline{\varphi})_i$ for all i with the coefficients from F. F is a field of constants. $\lceil x \rceil$ and $\lfloor x \rfloor$ are the two nearest integers such that $\lfloor x \rfloor \leq x \leq \lceil x \rceil$. $\lambda(v)$ is $o(\theta(v))$ if $\lambda(v)/\theta(v) \to 0$ for $v \to \infty$. $\lambda(v)$ is $O(\theta(v))$, respectively $\Omega(\theta(v))$ if there exist two constants $c_0 > 0$, $c_1 > 0$, such that $\lambda(v) < c_0\theta(v)$ for $|v| > c_1$, respectively $\lambda(v) > c_0\theta(v)$ for $|v| > c_1$. $\lambda(v) \sim \theta(v)$ if simultaneously $\lambda(v)$ is $O(\theta(v))$ and $\theta(v)$ is $O(\lambda(v))$. $\underline{X}$, $\underline{Y}$, $\underline{Z}$ are vectors of indeterminates, $x_i = (\underline{X})_i$, $y_i = (\underline{Y})_j$, $z_k = (\underline{Z})_k$. $x_i = y_j = z_k = 0$ unless $0 \leq i < I$, $0 \leq j < J$, $0 \leq k < K$. $f_{jk} \in F$, $f_{ijk} \in F$ for all i, j, k. If S is a set then $|S|$ is the cardinality of S.

Definition 2.1. A linear (arithmetic computational) problem is a set of linear forms,

$$\ell_k(\underline{Y}) = \sum_{j=0}^{J-1} f_{jk} y_j \quad \text{for } k = 0, 1, \ldots, K-1. \tag{2.1}$$

A bilinear (arithmetic computational) problem is a set of bilinear forms,

$$b_k(\underline{X}, \underline{Y}) = \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} f_{ijk} x_i y_j \quad \text{for } k = 0, \ldots, K-1. \tag{2.2}$$

Also a linear problem can be equivalently represented by the bilinear form, $b(\underline{Y}, \underline{Z})$, or by the matrix, μ such that

$$b(\underline{Y}, \underline{Z}) = \sum_{k=0}^{K-1} \ell_k(\underline{Y}) z_k, \quad (\mu)_{jk} = f_{jk} \text{ for all } j, k. \tag{2.3}$$

A bilinear problem can be equivalently represented by the trilinear form, $t(\underline{X}, \underline{Y}, \underline{Z})$, by the 3-dimensional tensor, τ, and by the matrices $\mu(\underline{X})$, $\mu(\underline{Y})$, $\mu(\underline{Z})$, such that for all

i, j, k

$$t(\underline{X},\underline{Y},\underline{Z}) = \sum_{k=0}^{K-1} b_k(\underline{X},\underline{Y}) z_k\,, \quad (\tau)_{ijk} = f_{ijk}\,, \tag{2.4}$$

$$\left(\mu(\underline{X})\right)_{jk} = \sum_{i=0}^{I-1} f_{ijk} x_i\,,$$

$$\left(\mu(\underline{Y})\right)_{ki} = \sum_{j=0}^{J-1} f_{ijk} y_j\,, \tag{2.5}$$

$$\left(\mu(\underline{Z})\right)_{ij} = \sum_{k=0}^{K-1} f_{ijk} z_k\,,$$

Here are some examples of linear and bilinear problems where m, n are given non-negative integers. We represent a linear problem by $b(\underline{X},\underline{Y})$ and μ and bilinear problems by $t(\underline{X},\underline{Y},\underline{Z})$ and $\mu(\underline{X})$. The transiton to the representations (2.1), (2.2) can be easily done with formulas (2.3)-(2.5).

Problem 2.1. (DFT, Discrete Fourier Transform in $n+1$ points).
$b(\underline{X},\underline{Z}) = \sum_{k=0}^{n} z_k \sum_{j=0}^{n} \omega^{jk} y_j$. $(\mu)_{jk} = \omega^{jk}$, ω is a primitive $(n+1)$-root of unity.

Problem 2.2. (CV, convolution of two vectors, or polynomial multiplication.
$t(\underline{X},\underline{Y},\underline{Z}) = \sum_{k=0}^{m+n} z_k \sum_{j=0}^{k} x_{k-j} y_j$. $\left(\mu(\underline{X})\right)_{jk} = x_{k-j}$, where

$$x_i = 0 \text{ if } i < 0 \text{ or } i > m\,, \qquad y_j = 0 \text{ if } j > n\,. \tag{2.6}$$

Remark 2.1. The matrix $\mu(\underline{X})$ of CV is a particular case of a general $(n+1)\times(m+n+1)$ Töeplitz matrix, $T(\underline{X})$; $\left(T(\underline{X})\right)_{jk} = x_{k-j}$, $j = 0, 1, \ldots, n$, $k = 0, 1, \ldots, m+n$. That particular case is defined by the following relations.

$$x_s = 0 \text{ for } s < 0 \text{ and for } s > m\,.$$

CV is the problem of the evaluation of the coefficients of the polynomial in λ, $q(\lambda)r(\lambda)$ where the coefficients of $q(\lambda) = \sum_{i=0}^{m} x_i \lambda^i$ and $r(\lambda) = \sum_{j=0}^{n} y_j \lambda^j$ are given.

Problem 2.3. (CCV, Cyclic convolution of vectors or multiplication of two n-the degree polynomials modulo $\lambda^{n+1} - 1$).

$$t(\underline{X},\underline{Y},\underline{Z}) = \sum_{k=0}^{n} z_k \left(\sum_{j=0}^{k} x_{k-j} y_j + \sum_{j=0}^{\bar{k}} x_{\bar{k}-j} y_j \right),$$

$$\left(\mu(\underline{X})\right)_{jk} = x_{k-j} \text{ if } j \le k\,, \qquad \left(\mu(\underline{X})\right)_{jk} = x_{\bar{k}-j} \text{ otherwise.} \tag{2.7}$$

Here $j, k = 0, 1, \ldots, n$, $\bar{k} = k + n + 1$. (2.7) defines $\mu(\underline{X})$ as the general $(n+1) \times (n+1)$ circulant matrix that is the general $(n+1) \times (n+1)$ Töeplitz matrix (cf. Remark 2.1), where $x_i = x_{i+n+1}$ for $i \le 0$.

Problem 2.4. (MM, matrix multiplication. $(I = J = K = n^2.)$

For this problem we represent the vectors $\underline{X}$, $\underline{Y}$, $\underline{Z}$ as the matrices X, Y, Z, so that $(\underline{X})_i = (X)_{\alpha\beta}$, $(\underline{Y})_j = (Y)_{\beta\gamma}$, $(\underline{Z})_k = (Z)_{\gamma\alpha}$, $i = \alpha n + \beta$, $j = \beta + \gamma n$, $k = \gamma n + \alpha$, $\alpha, \beta, \gamma = 0, 1, \ldots, n-1$.

$$t(\underline{X}, \underline{Y}, \underline{Z}) = \sum_{\alpha,\beta,\gamma=0}^{n-1} x_{\alpha\beta} y_{\beta\gamma} z_{\gamma\alpha} = \operatorname{Tr}(XYZ),$$

$$\left.\begin{array}{ll} (\mu(\underline{X})) = (X)_{\alpha\beta} & \text{if } j = \beta + \gamma n, k = \gamma n + \alpha \text{ for some } \alpha,\ \beta,\ \gamma, \\ (\mu(\underline{X})) = 0 & \text{otherwise.} \end{array}\right\} \qquad (2.8)$$

Here $\operatorname{Tr}(XYZ)$ is the trace of the matrix XYZ.

Remark 2.2. (2.8) defines $\mu(\underline{X})$ as the block-diagonal $n^2 \times n^2$ matrix where each of n diagonal blocks is the matrix X.

3. Linear and bilinear arithmetic algorithms.

Given a vector of input-variables (indeterminates), $\underline{V} = (v_1, \ldots, v_Q)$, and a field of constants, F, then an arithmetic algorithm A is defined as a sequence of arithmetic operations, $\langle a_1, \ldots, a_C \rangle$ such that

$$a_s = a_s(a_{g(s)}, a_{h(s)}), \quad g(s) < s, \quad h(s) < s, \quad s = 1, \ldots, C, \qquad (3.1)$$

each $a_s(u, w)$ is either $u + w$, or $u - w$, or uw, or u/w, $g(s)$, $h(s)$ are integers, $a_{1-i} = v_i$ for $i = 1, \ldots, Q$, $a_i \in F$ for $i \le -Q$. Given $\underline{a}^*$, a subset of the set $\{a_1, \ldots, a_C\}$, then A is said to evaluate $\underline{a}^*$ over F and to have the complexity C.

We will study arithmetic algorithms that evaluate given sets of linear forms, $\{\ell_k(\underline{Y})\}$, then $\underline{V} = \underline{Y}$, $Q = J$, cf. (2.1), or of bilinear forms, $\{b_k(\underline{X}, \underline{Y})\}$, then $\underline{V} = (\underline{X}, \underline{Y})$, $Q = I + J$, cf. (2.2). In those cases we consider the natural classes of linear and bilinear algorithms. An arithmetic algorithm (3.1) is called linear if for all s either

$$a_s(u, w) = uw \text{ and } a_{g(s)} \in F, \qquad (3.2)$$

or

$$a_s(u, w) = u + w \text{ or } a_s(u, w) = u - w. \qquad (3.3)$$

Operations (3.2) and (3.3) are called scalar multiplications and additions/subtractions (or operations $\pm$) respectively. Notice that for all s the a_s are linear forms in $\underline{V}$ in the case of linear algorithms.

An arithmetic algorithm (3.1) is called bilinear if it consists of the following four successive steps.

Step 1. Linear algorithm, A_1 where the set of input-variables is $\underline{X}$.

Step 2. Linear algorithm, A_2 where the set of input-variables is $\underline{Y}$.

Step 3. Multiplication algorithm, $A_3 = \langle a_1, \ldots, a_M \rangle$, where $a_s = a_{g(s)}a_{h(s)}$, $s = 1, \ldots, M$, all $a_{g(s)}$ are outputs of Step 1, and all $a_{h(s)}$ are outputs of Step 2. The operations of Step 3 are called nonscalar multiplications.

Step 4. Linear algorithm, A_4 where the set of input-variables is the set of outputs of Step 3. (If it is allowed to extend the sets of input-variables of each of Steps 1 and 2 to $(\underline{X}, \underline{Y})$, then Steps 1–4 are said to define a quadratic algorithm. Quadratic algorithms are not considered in this paper.)

4. The height of a linear form and the p-rank of a matrix.

In this section we define two auxiliary concepts (height and p-rank).

Definition 4.1. $H(\ell)$, the height of $\ell(\underline{Y})$, a linear form of $\underline{Y}$ (respectively of $b(\underline{X}, \underline{Y})$, a bilinear form of $\underline{X}$, $\underline{Y}$), with the coefficients from F, is the number of nonzero coefficients of the given form. H and $\tilde{H}$, the height and respectively the lower height of a (bi)linear problem are the maximum and respectively the minimum heights of the (bi)linear forms to be evaluated.

Notation. $C_A(\pm)$ is the number of $\pm$ operations involved in an algorithm A. $C_\ell(\pm)$ is the minimum $C_A(\pm)$ for all linear algorithms A that evaluate a linear form $\ell(\underline{Y})$.

The next obvious lemma enables us to estimate $C_\ell(\pm)$.

Lemma 4.1. *Let $\ell_1(\underline{Y})$, $\ell_2(\underline{Y})$ be linear forms of $\underline{Y}$ over F. Let $\ell(\underline{Y}) = f_1\ell_1(\underline{Y}) + f_2\ell_2(\underline{Y})$, where $f_1, f_2 \in F$. Then $H(\ell) \leq H(\ell_1) + H(\ell_2)$.*

Corollary 4.1, cf. [12]. $C_\ell(\pm) = H(\ell) - 1$.

The next three definitions will introduce another auxiliary concept, the p-rank of a matrix.

Definition 4.2. Two linear forms are p-neighbors if the height of their difference is at most p.

Definition 4.3. Given two sets of linear forms, $\{\ell_k(\underline{Y})\}$ and $\{\ell'_k(\underline{Y})\}$, and the two associated matrices of their coefficients, μ and μ', then μ and μ' are p-neighbors iff for each k the two linear forms $\ell_k(\underline{Y})$ and $\ell'_k(\underline{Y})$ are p-neighbors, or equivalently iff the matrix $\mu' - \mu$ has at most p nonzero entries in each column.

Definition 4.4. $r_p(\mu)$, the p-rank of a matrix, μ, is the minimum rank of *its* p-neighbors. $r_p(\mu(\underline{X}))$, the p-rank of a matrix $\mu(\underline{X})$, is $\max_{\underline{X}^0} r_p(\mu(\underline{X}^0))$.

Remark 4.1. Similar but different definitions could be obtained if "row" substitutes for "column". *$r_0(\mu)$ is the conventional rank of μ.*

As is obvious, an arbitrary $m \times m$ matrix, μ, has a p-neighbor with at least p zero-rows, so that $r_p(\mu) \leq m - p$. On the other hand, for a given r, the testing if $r_p(\mu) \leq r$ can be reduced to the solution of a finite number of systems, each of $(m-r)n$ algebraic equations for $(m-r)r + np$ variables. (Fix r candidates for r basic rows among the finite number of possible choices and try to express other $m - r$ rows as linear combinations of basic ones.) This suggests that $(m-r)n \leq (m-r)r + np$ and hence $(m+n-r)r \geq (m-p)n$. If $m = n$, $p = \gamma n$, $r = \beta n$, then $(2-\beta)\beta \geq 1 - \gamma$.

However, we do not see any way to turn the above preliminary estimates into the lower bounds on $r_p(\mu)$ and $r_p(\mu(\underline{X}))$ in the specific cases where μ and $\mu(\underline{X}))$ are the matrices of Problems 2.1–2.4. Those who look for hard and interesting problems are encouraged to try to prove or disprove *the following proposition.*

Proposition 4.1. There exists a constant α, $0 < \alpha < 1$ such that $p \leq n^\alpha$ implies that

$$r_p(\mu) = \Omega(n), \qquad r_p(\mu(\underline{X})) = \Omega(n)$$

if μ is the matrix of Problem 2.1 and $\mu(\underline{X})$ is the matrix of Problem 2.2 (where $m \sim n$) or 2.3. If $p \leq n^\alpha$ and $\mu(\underline{X})$ is the matrix of Problem 2.4 then

$$r_p(\mu(\underline{X})) = \Omega(n^2).$$

In Section 5.4 we show how to use Proposition 4.1 (if it is true) in our study of linear and bilinear algorithms, and how even weaker lower estimates for $r_p(\mu)$ and $r_p(\mu(\underline{X}))$ can be helpful.

5. Algorithms as digraphs and the lower bounds on the sum of the additive and logical complexities.

We partition this long section into four subsections.

5.1. (Bi)linear algorithm as a sequence of $\pm$ operations. Assume that the indeterminants $(\underline{X})_i$ for all i are replaced by their instances taken from F. Then, of course, each bilinear algorithm turns into a linear one and $C_A(\pm)$ does not increase. (We can assume that the instances, $(\underline{X})_i$, are always chosen such that $C_A(\pm)$ takes its maximum value.) Hence the lower bounds on the $C_{\min}(\pm)$ over the class of such linear algorithms are also the lower bounds on the $C_{\min}(\pm)$ over all bilinear algorithms for Problems 2.2–2.4. In this and in the next sections, the words "Problems 2.2-2.4" will be freely used as a shorthand for "the linear instances of Problems 2.2-2.4" where this does not cause confusion.

We will count only $\pm$ operations. Thus we rewrite an arbitrary linear algorithm, cf. (3.1)–(3.3), in a more convenient way for our analysis, that is, as $\langle a^+_{1-J}, a^+_{2-J}, \ldots, a^+_{C_A(\pm)} \rangle$, a sequence of linear forms in the input-variables, $\underline{Y}$, where

$$a^+_j = f_j a^+_{\nu(j)} + f'_j a^+_{\eta(j)}, \quad \nu(j) < j, \quad \eta(j) < j, \qquad f_j \in F, \quad f'_j \in F, \quad j = 1, 2, \ldots, C_A(\pm), \tag{5.1.1}$$

$$a^+_{-j} = y_j \quad \text{for } j = 0, 1, \ldots, J-1, \tag{5.1.2}$$

$$a^+_s \neq f\, a^+_q \quad \text{for all } f \in F, \quad s \neq q. \tag{5.1.3}$$

Notice that $C_A(\pm)$ equals the number of all a^+_j in (5.1.1) where $j \geq 1$.

Hereafter, we assume that the a^+_j is defined up to a constant nonzero factor $f \in F$, so that in fact each a^+_j is the set $\{f\, a^+_j, f \in F\}$ of proportional linear forms in $\underline{Y}$ where

$$H(a^+_j) > 1 \quad \text{iff } j \geq 1, \quad H(a^+_j) \leq H(a^+_{\nu(j)}) + H(a^+_{\eta(j)}) \quad \text{for all } j \geq 1.$$

5.2. Algorithms as digraphs partitioned into levels. Irregularity numbers. We associate A, an algorithm (5.1.1)–(5.1.3), with $D(A) = (V(A), E(A))$, an acyclic directed graph, where $V(A) = \{a^+_j, j = 1-J, 2-J, \ldots, C_A(\pm)\}$, $E(A) = \{(a^+_j, a^+_{\nu(j)}), (a^+_j, a^+_{\eta(j)}), j = 1, 2, \ldots, C_A(\pm)\}$. A vertex a^+_j has outdegree 2 if $j > 0$ and outdegree 0 if $j \leq 0$. The $C_A(\pm)$ is equal to the number of vertices of $V(A)$ of outdegree 2. This motivates our upcoming study of acyclic digraphs.

Definition 5.2.1. Given an acyclic digraph $D = (V, E)$ then a (directed) path in D of length $s \geq 0$ is a pair of chains of edges $\langle (v_0, v_1), (v_1, v_2), \ldots, (v_{s-1}, v_s) \rangle$ and of vertices $\langle v_0, v_1, \ldots, v_s \rangle$ where $(v_h, v_{h+1}) \in E$ for all h. (All paths are directed and have positive lengths.) A set $\tilde{V} \subseteq V$ circles $v \in V$ iff any directed path from v either enters $\tilde{V}$, or can be continued to $\tilde{V}$. The subset V^{*q} of $V^* \subseteq V$ consists of all vertices of V^* of outdegree q. Let $L_0(D)$ be a fixed (basic) subset of V. The shortest path level, $L_q(D)$ and the longest path level, $LL_q(D)$ are the two subsets of V such that $v \in L_q(D)$ (resp. $v \in LL_q(D)$) iff each shortest (resp. each longest) path from $L_0(D)$ to v consists of q edges. In this case $q = d(v)$ (resp. $q = r(v)$) is the distance (resp. the remove) from $L_0(D)$ to v. $d(D) = \max_{v \in V} d(v)$ is the depth of D. $p(D) = \max_{v \in V} r(v)$ is the profundity of D. $PL_q(D) = L_q(D) \setminus V^0$ and $PLL_q(D) = LL_q(D) \setminus V^0$ are the subsets of $L_q(D)$ and $LL_q(D)$ resp. They consist of the vertices of positive outdegrees. The set $\tilde{L}_q(D) = L_q(D) \cup_{s<q} \left(L_s(D)\right)^0$ is the level $L_q(D)$ extended by joining the vertices of outdegree 0 from the previous levels of D.

Now we can write the following basic relation,

$$C_A(\pm) = \sum_{q=0}^{d(D)-1} |PL_q(D)| = \sum_{q=0}^{p(D)-1} |PLL_q(D)| \qquad (5.2.1)$$

if $D = D(A)$ represents a linear algorithm A.

Remark 5.2.1. We always choose the set of outputs of A as the basic subset $L_0(D)$ if $D = D(A)$.

We start with lower estimates for $|\tilde{L}_q(D)|$. Such estimates will follow as a corollary from the next two lemmas.

Lemma 5.2.1. *Given an acyclic digraph $D = (V, E)$ with a basic subset $L_0(D) \subseteq V$, a positive integer q, and a vertex $v \in L_0(D)$. Then the set $\tilde{L}_q(D)$ circles v.*

Remark 5.2.2. If $\tilde{L}_q(D)$ had been defined as $LL_q(D) \cup_{s<q} \left(LL_s(D)\right)^0$ then Lemma 5.2.1 would not have held. (For some digraphs D, $\left(L_0(D)\right)^0$ and $LL_1(D)$ are empty, $LL_2(D)$ is not.)

Lemma 5.2.2. *Let a digraph $D = D(A) = \big(V(A), E(A)\big)$ represent a linear algorithm A; cf. (5.1.1)-(5.1.3). For an integer j and for a set of integers S, all from the closed interval $\big(1 - J, C_A(\pm)\big)$, let the subset $\{a_s^+, s \in S\}$ of $V(A)$ circle $a_j^+ \in V(A)$. Then $a_j^+ = \sum_{s \in S} f_s a_s^+$, $f_s \in F$ for all s.*

Corollary 5.2.1. *Let a digraph $D = D(A) = \big(V(A), E(A)\big)$ represent a linear algorithm A for a linear problem with matrix μ, cf. Remark 5.2.1. Then*

$$|\tilde{L}_q(D)| \geq r_0(\mu) \quad \textit{for } q = 0, 1, \ldots, d\big(D(A)\big) - 1\,.$$

Here $r_0(\mu)$ is the rank of μ.

Proof. $r_0(\mu)$ is the number of linearly independent outputs. That number is not less than $|\tilde{L}_q(D)|$ by Lemmas 5.2.1 and 5.2.2. ∎

Next we estimate $C_A(\pm)$ using (5.2.1) and the following definition.

Definition 5.2.2. An acyclic digraph, $D = (V, E)$ is shortest (res. longest) path regular one iff all its vertices of outdegree 0 belong to the level $L_{d(D)}(D)$ (resp. $LL_{p(D)}(D)$). The two numbers, $ir^*(D) = \sum_{v \in V^0} ir^*(v)$, $ir(D) = \sum_{v \in V^0} ir(v)$, are the shortest and resp. the longest path irregularity numbers of D. Here $ir^*(v) = d(D) - d(v)$, $ir(v) = p(D) - p(v)$.

Remark 5.2.3. D can always be transformed into a shortest (resp. longest) path regular acyclic digraph by joining the path defined by the chain of vertices $\langle v_h(v), h = 0, 1, 2, \ldots, s\rangle$ to each $v \in V^0$. Here $v_0(v) = v$ and s is the length of the chain, $s = ir^*(v)$ (resp. $s = ir(v)$). Totally $ir^*(D)$ (resp. $ir(D)$) new vertices and as many new edges are to be joined to D during the regularization process.

Combining Definition 5.2.2, Corollary 5.2.1, and Equation (5.2.1), we obtain the following estimate.

Corollary 5.2.2. *Under the conditions of Corollary 5.2.1,* $C_A(\pm) + ir^*(D(A)) \geq r_0(\mu)d(D(A))$.

The latter estimate itself gives little guidance to the designers of the fast linear algorithms. Indeed, $C_A(\pm) = O(r_0(\mu))$ implies that $r_0(\mu)d(D(A)) - ir^*(D(A)) = 0(r_q(\mu))$ but in general $d(D(A))$ can be as small as 2 for any linear problem. (Indeed, given an algorithm A (cf. (5.1.1)) then for $a \in L_0(D(A))$, $\tilde{f} \in F$, $j > 0$, $f_j \neq 0$, substitute the next three $\pm$ steps for (5.1.1): $\tilde{a}_{\nu(j)} = a^+_{\nu(j)} + \tilde{f}_j a$, $\tilde{a}_j = f_j \tilde{a}_{\nu(j)} + f'_j a^+_{\eta(j)}$, $a^+_j = \tilde{a}_j - f_j \tilde{f}_j a$. Choose a, $\tilde{f}_j$ such that $\tilde{a}_s \neq f\, a^+_q$ for all q and for $s = j$, $s = \nu(j)$.)

In the next subsection we present a more meaningful lower estimate.

5.3. Ordered levels, disorder numbers and lower bounds.

Definition 5.3.1. The levels of $D = (V, E)$ are ordered up to the level $LL_q(D)$ if $r(v) = r(u) + 1$ for any edge $(u, v) \in E$ such that $r(v) \leq q$. The levels of D are ordered if they are ordered up to the level $LL_q(D)$ where $q = p(D)$. $do_q(D)$ is the disorder number of $\{LL_s(D), s = 0, 1, \ldots, q\}$ if

$$do_q(D) = \sum_v do(v), \quad do(v) = \max_u do(u, v), \quad do(u, v) = r(v) - r(u) - 1. \qquad (5.3.1)$$

Here the maximization is for all u such that

$$(u, v) \in E, \qquad (5.3.2)$$

and the summation is for all v such that $r(v) \leq q$. $do(D) = do_{p(D)}(D)$ is the disorder number of D. $do(u,v)$ is the disorder generated by the edge (u,v) and $do(v)$ is the disorder around v.

The next two simple lemmas and corollary indicate the way to apply the digraphs with the ordered levels to our main problem.

Lemma 5.3.1. *$L_s(D) = LL_s(D)$ for $s = 0, 1, \ldots, q$ if the levels of D are ordered up to the level $LL_q(D)$. In particular, if $q = p(D)$ then $p(D) = d(D)$, $ir(D) = ir^*(D)$.*

Lemma 5.3.2. *Under the conditions of Corollary 5.2.1, $p(D(A)) \geq \log H$ where H is the height of the linear problem solved by A.* (To prove, apply Lemma 4.1.)

Corollary 5.3.1. *Let a digraph $D = D(A)$ with ordered levels represent a linear algorithm A for a linear problem with matrix μ. Let H designate the height of the problem (cf. Definition 4.1). Then*

$$d(D(A)) \geq \log H\,, \quad C_A(\pm) + ir(D(A)) \geq r_0(\mu) \log H\,.$$

(Corollary 5.3.1 follows from Lemmas 5.3.1, 5.3.2, and Corollary 5.2.2.)

Next we will show how to order the levels of an arbitrary acyclic digraph and will estimate the cost of such ordering.

Lemma 5.3.3. *For any $q \geq 0$ an acyclic digraph $D = (V, E)$ can be transformed into a digraph $\tilde{D}_q = (\tilde{V}_q, \tilde{E}_q)$ whose levels are ordered up to the level $LL_q(\tilde{D}_q)$ and such that*

$$\tilde{V}_q^i = V^i \quad \text{for } i \neq 1\,, \quad \tilde{V}^1 \supseteq V^1\,, \quad |V_q^1| = |V^1| + do_q(D)\,. \tag{5.3.3}$$

In particular, $\tilde{D}_{p(D)} = \tilde{D} = (\tilde{V}, \tilde{E})$ has ordered levels and

$$\tilde{V}^i = V^i \quad \text{for } i \neq 1\,, \quad \tilde{V}^1 \supseteq V^1\,, \quad |\tilde{V}^1| = |V^1| + do(D)\,. \tag{5.3.4}$$

Proof. Let $r(v) \leq q$. Let $U(v)$ designate the set of all vertices $u \in V$ such that (5.3.2) holds and $E(v)$ designate the set of edges (u,v) of E with the head v. Then substitute the following subdigraph $\tilde{D}(v) = (\tilde{V}(v), \tilde{E}(v))$ for the $E(v)$; cf. (5.3.1).

$$\tilde{V}(v) = \{\tilde{v}_h(v)\,,\ h = 1, 2, \ldots, do(v)\}\,.$$

$$\tilde{E}(v) = \{(\tilde{v}_s(v)\,,\ \tilde{v}_{s-1}(v)) \quad \text{for } s = 1, 2, \ldots, do(v)\,, \quad (u, \tilde{v}_{do(u,v)}(v)) \quad \text{for all } u \in U(v)\}$$

where $\tilde{v}_0(v) = v$.

The desired digraph $\tilde{D}_q = (\tilde{V}_q, \tilde{E}_q)$ results if such substitutions are done for all $v \in V$ such that $r(v) \leq q$. Indeed, (5.3.3) is easily verified, as well as the following property: for each edge $(u, v) \in E$ such that $r(v) \leq q$ in D, the shortest directed path in $\tilde{D}_q$ from u to v has the length $do(u, v) + 1$ and has only the vertices of outdegree 1 except u and v. This implies that $r(\tilde{v}) = r(\tilde{u}) + 1$ for all $(\tilde{u}, \tilde{v}) \in \tilde{E}_q$ such that $r(\tilde{v}) \leq q$. ∎

Remark 5.3.1. The remove of a common vertex v of V and $\tilde{V}$ is the same in D and $\tilde{D}$. By the virtue of Lemma 5.3.1, it equals $d(v)$ in $\tilde{V}$. In particular, it follows that

$$p(D) = p(\tilde{D}) = d(\tilde{D}) \geq d(D), \quad ir(D) = ir(\tilde{D}) = ir^*(\tilde{D}) \geq ir^*(D).$$

Combining the transformations of D of Remark 5.2.3 and of the proof of Lemma 5.3.3, we turn D into $\overline{D}$, a longest path regular digraph with ordered levels. By the virtue of Lemma 5.3.1 and Remark 5.3.1, $\overline{D}$ is also a shortest path regular digraph. Combining Remarks 5.2.3 and 5.3.1, Corollary 5.3.1, and Lemma 5.3.3 we obtain the following estimate.

Theorem 5.3.1. *Let a digraph $D = D(A)$ represent a linear algorithm A associated with matrix μ. Let H be the height of the problem. Then*

$$C_A(\pm) + do(D(A)) + ir(D(A)) \geq r_0(\mu) \log H . \tag{5.3.5}$$

As is easily verified,

$$r_0(\mu) \log H = (n+1) \log(n+1) \quad \text{for Problems 2.1, 2.3,} \tag{5.3.6}$$

$$r_0(\mu) \log H = (m+n+1) \log \max(m+1, n+1) \quad \text{for Problem 2.2,} \tag{5.3.7}$$

$$r_0(\mu) \log H = 2n^2 \log n \quad \text{for Problem 2.4.} \tag{5.3.8}$$

On the one hand, relations (5.3.5)-(5.3.7) enable us to narrow the search of new faster algorithms for DFT, CV, CCV. On the other hand, even if such algorithms exist, they must have some deficiencies in their logical structure that are quantitatively represented by the sum $do(D(A)) + ir(D(A))$. The increase of $ir(D(A))$ might seem a lesser evil. This motivates our next study of the lower bounds on $C_A(\pm) + do(D(A))$ based on the two simple lemmas presented in the beginning of the next subsection.

5.4. Lower bounds on $C_A(\pm) + do(D(A))$ in terms of the p-ranks of the matrix of the problem.

Lemma 5.4.1. *Let all vertices of an acyclic digraph D have outdegrees 0, 1, or 2. Let $v \in L_0(D)$ and let $q \geq 0$ be an integer. Then for some $s \leq 2^q$ there exist s vertices from $\tilde{L}_q(D)$ (cf. Definition 5.2.1) that circle v.*

Proof. Apply Lemma 5.2.1. The bound 2 on the outdegrees implies that at most 2^q paths from v to $\tilde{L}_q(D)$ exists. ∎

Lemma 5.4.2. *Let q be a nonnegative integer and $D = D(A)$ represent a linear algorithm, A. Let $v \in L_0(D)$ be an output of A (which is a linear form in $\underline{Y}$). Then there exists an h-neighbor of v (cf. Definition 4.2) that can be represented as a linear combination of g vertices of positive outdegrees in D which are elements of $L_q(D)$ and linear forms in $\underline{Y}$, such that $\log(g+h) \leq q$. If the height of v (cf. Definition 4.1) is greater than 2^q then $g \geq 1$.*

Proof. Apply Lemmas 5.2.2 and 5.4.1. Represent v as $\sum_{i=1}^{g+h} f_i v_i$ where (cf. Definition 5.2.1) $v_i \in \cup_{s=0}^{q}(L_s(D))^0$ for $i = 1, \ldots, h$, $v_i \in PL_q(D)$ for $i = h+1, \ldots, h+g$.

Recall that all v_i and v are linear forms in $\underline{Y}$ with coefficients from F and notice that the height of v_i equals 1 for $i = 1, \ldots, h$. Hence $v - \sum_{i=1}^{h} f_i v_i = \sum_{i=h+1}^{h+g} f_i v_i$ is an h-neighbor of v. If $g = 0$ the height of v equals $h \leq 2^q$. ∎

Corollary 5.4.1. *Let a digraph $D = D(A)$ represent a linear algorithm for a problem with matrix μ (cf. Remark 5.2.1). Then for any integer $s \geq 0$,*

$$|PL_s(D)| \geq r_{h(s)}(\mu), \tag{5.4.1}$$

$$h(s) = 2^s - 1 \quad \textit{if the heights of all outputs of } A \textit{ are greater than } 2^s, \tag{5.4.2}$$

$$h(s) = 2^s \quad \textit{otherwise.} \tag{5.4.3}$$

Remark 5.4.1. For $s = 0$ the estimate (5.4.1) can be improved as follows,

$$|PL_0(D)| \geq c(\mu). \tag{5.4.4}$$

Here and hereafter $c(\mu)$ designates the number of nonproportional outputs of the heights greater than 1 or equivalently the number of nonproportional columns of μ with more than 1 nonzero entries. If $m \geq 1$, $n \geq 1$ then $c(\mu) = n + 1$ for Problems 2.1, 2.3, $c(\mu) = m + n - 1$ for Problem 2.2, and $c(\mu) = n^2$ for Problem 2.4. Bound (5.4.1) can also be strengthened for $s > 0$. We leave this possibility as a challenge to the reader.

Let a digraph $D(A)$ represent a linear algorithm A; cf. (5.1.1)-(5.1.3). We can apply the transformation of $D(A)$ from Lemma 5.3.3 to order the levels of $D(A)$. Then combining Equation (5.2.1), Lemma 5.3.3, Corollary 5.4.1, and the bound (5.4.4) gives the following estimate.

Theorem 5.4.1. *Let a digraph $D = D(A)$ represent a linear algorithm A for the problem with a matrix μ (cf. Remarks 5.2.1, 5.4.1). Then (cf. (5.4.2), (5.4.3)) for all q*

$$C_A(\pm) + do_q(D(A)) \geq c(\mu) + \sum_{s=1}^{q} r_{h(s)}(\mu) + \sum_{s=q+1}^{p(D)-1} |PLL_s(D)| . \qquad (5.4.5)$$

In particular, for $q = p(D) - 1$,

$$C_A(\pm) + do(D(A)) \geq c(\mu) + \sum_{s=1}^{p(D)-1} r_{h(s)}(\mu) . \qquad (5.4.6)$$

Those readers who believe that Proposition 4.1 is true (why not?) will probably be happy with the following result (cf. (5.4.6).

Corollary 5.4.2. *If Proposition 4.1 holds then for any linear algorithm A for Problems 2.1–2.3, $C_A(\pm) + do(D(A)) = \Omega(n \log n)$ (assuming that $m \sim n$ in the case of Problem 2.2) and for any linear algorithm A for MM (Problem 2.4), $C_A(\pm) + do(D(A)) = \Omega(n^2 \log n)$. Here $D(A)$ is the digraph that represents A.*

Thus a designer of an algorithm for Problems 2.1-2.3 where $m = n$, $C_A(\pm) + do(D(A)) = o(n \log n)$ must at least disprove Proposition 4.1.

In fact, some linear lower bounds on $C_A(\pm) + do(D(A))$ that exceed $K + Q$ can be established immediately. For instance, the following results can be proven, see [20].

Theorem 5.4.2. *Let A be an arbitrary linear algorithm for one of the Problems 2.1–2.4. Then the following lower bounds on $C_A(\pm) + do(A)$ hold: $2n + 1.1q(n+1) - 2^8$ for DFT, $m + 2.33n + 0.3\min(m,n) - 2^8$ for CV (where $m \geq 2$, $n \geq 2$), $3.1(n+1) - 2^8$ for CCV, $3.5n^2 - 10.5n + 8$ for MM ($n \geq 2$). Here $q(s)$ designates the maximum prime that divides an integer s.*

In the next section we reconsider the problem of lower bounds on $C_A(\pm)$.

As follows from (5.3.5)-(5.3.8),

$$C_A(\pm) = \Omega(K + Q) \log(K + Q)$$

in the case of algorithms for DFT, CV, CCV and MM whose digraphs are regular and have ordered levels. When the paper had been written, Thomas Spencer, a graduate student at Stanford University, Stanford, California, noticed that the latter bound holds even if the regularity requirement is deleted. He also raised the questions if the term $ir(D(A))$ in (5.3.5) can be deleted or at least if in the cases of DFT, CV, CCV and MM,

$$K + Q = o(C_A(\pm) + do(D(A)).$$

These questions remain open. Unsuccessfully trying to answer them, the present author noticed however that for an arbitrary linear algorithm A for a problem of the height H associated with the matrix μ the next estimate immediately follows from our construction of Sections 5.1–5.3.

$$C_A(\pm) + do(\tilde{D}(A)) \geq r_0(\mu) \log \tilde{H}. \tag{5.4.7}$$

(Consequently $C_A(\pm) + do(\tilde{D}(a)) \geq Q \log Q$ for DFT, CV, where $m = n$, CCV and MM.) Here $\tilde{D} = \tilde{D}(A)$ is the acyclic digraph obtained from $D(A)$ by reversing the directions of all edges of $D(A)$ and the basic set $L_0(\tilde{D})$ is the set of all vertices of $\tilde{D}$ associated with the input-variables of A or equivalently having indegree 0 in $\tilde{D}$. In this case the basic relation,

$$C_A(\pm) = \sum_{q=1}^{d(\tilde{D})} |L_q(\tilde{D})| = \sum_{q=1}^{p(\tilde{D})} |LL_q(\tilde{D})|,$$

is simpler than (5.2.1) and also we can substitute $L_q(\tilde{D})$ for $\tilde{L}_q(D)$ in Lemma 5.2.1. This allows us to dispense with the regularization of $\tilde{D}$. The meanings of the numbers $do(\tilde{D}(A))$ and $do(D(A))$ are similar although those numbers can be different.

6. Lower bounds on $C(\pm)$ in terms of the rank of determinant.

Hereafter $L(\underline{X}, F)$ designate the set of all linear forms in $\underline{X}$ with coefficients from a field F. (The set $L(\underline{X}, F)$ includes the elements of F.)

Definition 6.1. If $P_n(\underline{X}) = P_n(\underline{X}_1, \ldots, \underline{X}_n)$ is an n-linear form in n vectors of indeterminants, $\underline{X}_1, \ldots, \underline{X}_n$ then $R(P_n(\underline{X}))$, the polylinear rank of $P_n(\underline{X})$, equals (cf. [1, p. 488] or [17]) the minimum integer R such that

$$P_n(\underline{X}) = \sum_{g=1}^{R} \prod_{h=1}^{n} L_{gh}(\underline{X}_h), \quad L_{gh}(\underline{X}_h) \in L(\underline{X}_h, F). \tag{6.1}$$

If, more generally, $P_n(\underline{X})$ is any polynomial of degree n in $\underline{X}$ then $r(P_n(\underline{X}))$, the rank of $P_n(\underline{X})$, is the minimum integer r such that

$$P_n(\underline{X}) = \sum_{g=1}^{r} \prod_{h=1}^{n} L_{gh}(\underline{X}), \quad L_{gh}(\underline{X}) \in L(\underline{X}, F). \tag{6.2}$$

For a matrix $\mu = \mu(\underline{X})$ with the entries from $L(\underline{X}, F)$,

$$r(\mu) = \max_{m \in D} r(m) \tag{6.3}$$

(where $D = D(\mu(\underline{X}))$ is the set of all minors of $\mu(\underline{X})$) is the rank of the bilinear problem associated with $\mu(\underline{X})$.

Equations (6.1) and (6.2) imply that

$$r(P_n(\underline{X})) \leq R(P_n(\underline{X})). \tag{6.4}$$

The $R(P_2(\underline{X}_1, \underline{X}_2))$ is equal to the rank of the matrix associated with a bilinear form $P_2(\underline{X}_1, \underline{X}_2)$. $R(P_3(\underline{X}_1, \underline{X}_2, \underline{X}_3))$ is equal to the multiplicative complexity of the three bilinear computational problems associated with $P_3(\underline{X}_1, \underline{X}_2, \underline{X}_3)$ (cf. [18, 19]).

Next we will modify the construction of [16] to show how the additive complexity of the bilinear problem associated with a matrix, μ, is related to the rank $r(\mu)$ (cf. (6.3)).

In Section 2 we associated linear forms in $\underline{Y}$ (whose coefficients are from $L(F, \underline{X})$) with the column-vectors of their coefficients. An s-tuple of such vectors forms a $J \times s$ matrix. In particular, the sequence of linear forms a_j^+, $j = 1 - J, 2 - J, \ldots, C_A(\pm)$ (cf. (5.1.1)), as well as the associated column-vectors of their coefficients, represents an arbitrary bilinear algorithm. The matrix $\mu(q)$ that consists of the first $J + q$ columns represents the algorithm up to its qth $\pm$ operation.

The set of the matrices $\{\mu(q), q = 1 - J, \ldots, C_A(\pm)\}$ must satisfy the following properties.

i) $\mu(0)$ is the $J \times J$ identity matrix.

ii) $\mu(q+1) = (\mu(q) \mid \underline{V}(q+1))$ for $q = 0, 1, \ldots, C_A(\pm) - 1$ where $\underline{V}(j) = f_j \underline{V}(\nu(j)) + f_j \underline{V}(\eta(j))$, $f_j \in F$, $f'_j \in F$, $\nu(j) < j$, $\eta(j) < j$, $\underline{V}(s)$ is the $(J+s)$th column-vector of $\mu(s)$; cf. (5.1.1).

iii) The matrix of the given bilinear problem $\mu(\underline{X})$ is a submatrix of $\mu(C_A(\pm))$.

Now we observe (cf. [16]) that the property ii) implies the next identities in $\underline{X}$ for all $q \geq 0$, $m(q+1) = f\,m(q) + f'\,m'(q)$. Here $m(q+1)$ is an arbitrary minor of $\mu(q+1)$, $f \in F$, $f' \in F$, and $m(q)$, $m'(q)$ are some minors of $\mu(q)$. Hence (cf. (6.3))

$$r(\mu(q+1)) \leq 2r(\mu(q)) \quad \text{for } q = 0, 1, \ldots, C_A(\pm) - 1. \tag{6.5}$$

As is obvious,

$$r(\mu(0)) = 1 . \tag{6.6}$$

$$r(\mu(C_A(\pm))) \geq r(\mu(\underline{X})) . \tag{6.7}$$

Equations (6.5)-(6.7) imply the following result.

Theorem 6.1. *Given a bilinear algorithm, A, for the bilinear problem defined by a matrix $\mu = \mu(\underline{X})$. Then $C_A(\pm) \geq \log r(\mu(\underline{X}))$.*

Remark 6.1. If one proved that

$$I = o(\log r(\mu(\underline{X}))) \tag{6.8}$$

for some $\mu(\underline{X})$, a fixed $I \times I$ matrix with the entries from $L(F, \underline{X})$ $(\underline{X} = (x_0, \dots, x_{I-1}))$ (for instance, for the $I \times I$ general Töeplitz matrix; cf. Remark 2.1) then Theorem 6.1 would imply a nonlinear lower bound on $C_A(\pm)$ for any algorithm A for the bilinear problem defined by that matrix $\mu(\underline{X})$. However inequality (6.4) and the bound $\log R(\text{per } \mu(\underline{X})) \leq I$ known for an arbitrary $I \times I$ matrix $\mu(\underline{X})$ (H. J. Ryser, cf. [1, p. 497]) convince that it is very hard to prove (6.8) if possible at all. Nevertheless we conclude with the following challenging problem: find an appropriate modification of the definition of $r(\mu(\underline{X}))$ such that Equations (6.5)-(6.8) hold.

Acknowledgments. I wish to thank Dr. D. Dolev for his helpful comments, particularly for his suggestion to omit the long proof of Theorem 5.4.2 and Thomas Spencer for his comments reported on in Section 5.4. I am also grateful to Kathy Lunetta and Phyllis Winkler for typing the first draft and the revision of this paper.

References.

1. D. E. Knuth, *The Art of Computer Programming: Seminumerical Algorithms* **2**, Addison-Wesley (1981).
2. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms,* Addison-Wesley (1976).
3. A. Borodin and I. Munro, *The Computational Complexity of Algebraic and Numeric Problems*, American Elsevier (1975).
4. S. Winograd, *Arithmetic Complexity of Computations,* SIAM (1980).
5. V. Ya. Pan, "The Bit-Operation Complexity of Approximate Evaluation of Matrix and Polynomial Products Using Modular Arithmetic," to appear in *Computers and Mathematics (with Applications).*
6. V. Ya. Pan, "New Combinations of Methods for the Acceleration of Matrix Multiplication," *Computers and Mathematics (with Applications)* **7**, 1 (1981), 73–125.
7. D. Coppersmith and S. Winograd, "On the Asymptotic Complexity of Matrix Multiplication," IBM T. J. Watson Reseach Center (May 1981).
8. A. L. Toom, "The Complexity of a Scheme of Functional Elements," *Doklady Akad. N[illegible] SSR* **150** (1963), 496–498. (Translated *Soviet Math.* **3** (1963), 714–716.)
9. [illegible]. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of [illegible]lex Fourier Series," *Math. Comp.* **19** (1965), 297–301.
10. [illegible]. Rader, "Discrete Fourier Transform when the Data Samples are Prime," *Proc. [illegible]EE* **56** (1968), 1107-1108.
11. C. M. Fiduccia and Y. Zalstein, "Algebras Having Linear Multiplicative Complexities," *Journal ACM* **24**, 2 (1977), 311–331.
12. E. G. Belaga, "Some Problems in the Computation of Polynomials," *Doklady Akad. Nauk SSSR* **123** (1958), 775–777 (in Russian).
13. V. Ya. Pan, "On Some Methods of Computing Polynomial Values," *Problemy Kibernetiki* **7** (1962), 21–30. (Translated *Problems of Cybernetics, USSR* **5** (1962), 20–30.)
14. V. Ya. Pan, "Methods for Computing Polynomials," (in Russian), Ph.D. Thesis, Dept. of Mechanics and Mathematics, Moscow State University (1964).
15. D. G. Kirkpatrick and Z. M. Kedem, "Additional Requirements for Rational Functions," *SIAM J. on Comp.* **6**, 1 1977), 188–199.
16. J. Morgenstern, "Note on a Lower Bound of the Linear Complexity of the Fast Fourier Transform," *Journal ACM* **20**, 2 (1973), 305–306.
17. N. Bourbaki, *Algèbre* **2**, Hermann, Paris, A2 (1970), 111–112.
18. V. Strassen, "Evaluation of Rational Functions," in *Complexity of Computer Computations* (edited by R. E. Miller and J. W. Thatcher), Plenum Press, N.Y. (1972).

19. V. Ya. Pan, "On Schemes for the Computation of Products and Inverses of Matrices," (in Russian), *Russian Math. Surveys* 27, 5 (1972), 249–250.
20. V. Ya. Pan, to appear.

FILMED

4-8